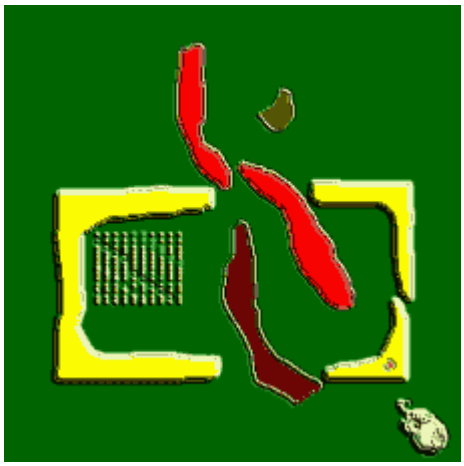
engene
Gene-Expression Data Processing and Exploratory Data Analysis

Computer Architecture Dept.
University of Malaga (Spain)
Biocomputing Unit
National Center for Biotechnology (Spain)



engene™

Welcome

Welcome to **engene™**, a versatile, web-based and platform independent exploratory data analysis tool for gene expression data that aims at storing, visualizing and processing large sets of expression patterns. engene (standing for Gene Engine) integrates a variety of analysis tools for visualizing, pre-processing and clustering expression data. The system includes different filters and normalization methods as well as an efficient treatment of missing data. The clustering algorithms included in the system range from the classical partitional and hierarchical methods, to the complex fuzzy ones, including: k-means, HAC, Fuzzy c-means and Kernel c-means. Linear and non-linear projection methods such as PCA, Sammon, and different variants of Self-Organizing Maps (classical, Fuzzy and Probabilistic) are also provided, including a completely novel SOM strategy aiming at producing truly quantitative Self-Organizing maps. Novel strategies for data pre-processing, gene and sample clustering and feature selection are also incorporated. Additionally, a Java suite for interactive Self-organizing Maps and partitional clustering is also included in the system. This tool enables the analysis of large sets of gene expression data in an easy and transparent manner, allowing the analysis of the outcome of different pre-processing and clustering methods at the same time. Free access to this tool is available upon request

engene™ is a trademark of Integromics

Integromics
Information Technologies for Life Sciences

www.integromics.com

engene: User Manual

About this document

This document concerns with some general but important terminology that must be mastered to fully understand the **engene** application following technical and training documents.

Cluster and classification analysis can be performed on many different types of data sets and in many application domains, such as engineering, biology, medicine or marketing, that have contributed to the development of novel approaches. Although procedures and definitions in this document are generic and valid with independence of the application domain, most of the examples are focused on clustering and classification of *gene expression data*. This is the field for which **engene** has been specially optimised, even when this application can be used for general cluster analysis.

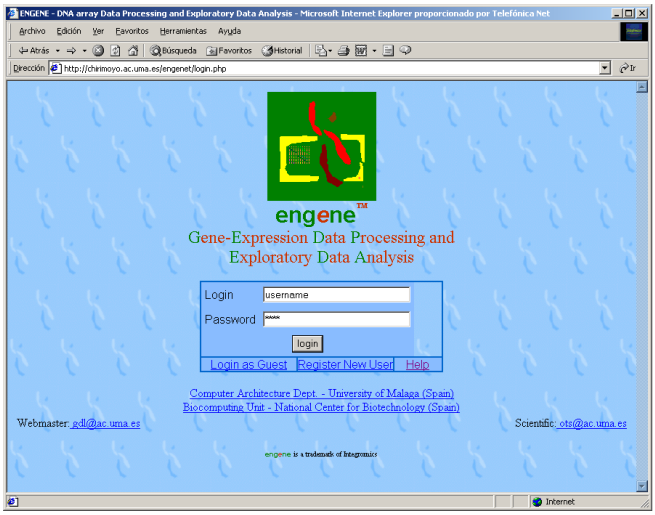
The two key applications gene-expression data collections are classification and clustering. Classification, also known as *discriminant analysis* or *supervised learning*, places an unknown object (gene or experiment) in one and only one of the *a priori* defined groups. By contrast in clustering analysis, also known as *unsupervised learning*, the classes are unknown a priori and the objective is to determine these classes from the data themselves, this is to say, identify genes (or experiments) with similar expression patterns from which their involvement in related biological processes may be deduced.

In this sense, **engene** is a discovering tool. It may reveal associations and structure in data which, though not previously evident, nevertheless are sensible and useful once found. The results of cluster analysis may contribute to the definition of a formal classification scheme, such as a taxonomy; or suggest statistical models to describe populations; or indicate rules for assigning new cases to classes for identification and diagnostic purposes; or provide measures of definition, size and change in what previously were only broad concepts; or find exemplars to represent classes.

Scope: This document is devoted to give an overview on **engene** application. This is aimed only as general information about the way in which data are up-loaded to the application, pre-processed and explored through the use of several data analysis tools. This document describe in general terms the available operations, their descriptions, and their inter-relations. A more detailed description about each option is available in the on-line help inside the web-application.

Login Page

The login page is the system entrance door. The main reason of this page is users identification and authorization. A user is identified by means of a **login** and a **password**. When the system has checked the goodness of these two words, the user is driven to his home directory (see Directory List); otherwise, the system entrance is denied and the user stays in the login page.


ENGENE - DNA array Data Processing and Exploratory Data Analysis - Microsoft Internet Explorer proporcionado por Telefónica Net
engene™
Gene-Expression Data Processing and Exploratory Data Analysis
Login
username
Password
login
Login as Guest
Register New User
Help
Computer Architecture Dept. - University of Malaga (Spain)
Biocomputing Unit - National Center for Biotechnology (Spain)
Webmaster: gdl@ac.uma.es
Scientific: ots@ac.uma.es
engene is a trademark of Integromics

Name	(Required)
Surname	(Required)
Institution	(Required)
Department	(Required)
Title	
Position	
Address	
Telephone	(Required)
Fax	
Email address	(Required)
Proposed login	(Required)
Proposed password	(Required)

The username (user identification) is a unique word, that identifies the user, and allows to assign different access controls (on data as well as on the application options). The password is a matter of security; it is encrypted and should not be shared by other users. Logins and Passwords are assigned by the Application Administrator. Once a user has written his identification name and his password, he must press the **Login** button.

A user can enter the system as a guest, by clicking on **Login As Guest**. In this case he will have more restricted options: he will be able to read data and to view them, but he will not be able to modify or process them. This option is specially suitable for an initial training purpose.

To enter to system as a standard user, a user has to register as a new user, the first time. When clicking **Register New User** he is driven to a *register form* where he is asked for his data. Based on these data, the system administrator will proceed to register the user (or decline the process when not appropriated)

Directory list

The **Directory list** page shows a files directory. This directory belongs to the user in **User name**. The **User name** links to the user home directory. The current directory path is shown at **Current directory**. This path is organized into click-able subdirectories. The user available free space is shown on the right, in **Quota left**. Once this available free space has run out, the user will not be able to do anything except delete or rename actions.

Contents

The files list is shown at the centre of the page. For each file, there are a file type icon, a file name, a file size and a file creation date. The following table shows the different file types recognized by **engene**:

	Generic file type		Data file		Codebook file
	Map file		Fuzzy Codebook file		Fuzzy Map file
	Distances histogram file		Values histogram file		Hierarchic tree file
	Main features file		Information file		Progress execution file
	Silhouette file		Transaction file		Association Rules file
	Sammon file		Directory		

engene implements a file-based navigation philosophy. It is necessary to select a file to make any process with it. Once it has been selected, the information related to this file (file-type dependent) is shown in a new page, with all the possible operations that can be realized on it. To obtain information about the different files page and about the operations that can be realized on them, just use the links of the previous table.

File Types

A **generic file** is a file with a none **engene** extension. In general, it contains text information.

Data File. A data file contains a list of vectors (data), all of the same dimension (number of variables). Moreover, a file may contain some metadata, arranged in arrays labels, variables labels and global labels. A more detailed data file description is shown above at Data File Format

Codebook File. A codebook file contains a data (vectors) classification. This arrangement is made of outstanding vectors, the code vectors. Each vector represents a classification class. In a codebook file, there is no relation between these code vectors. There is also an additional information, that associated the original data file with the classification. Each original vector might have been assigned to a code vector. To see that, for each code vector, there is a list of the indexes of the source data file original vectors . Since indexes are used, instead the vectors themselves, some operations over this file will be impossible without the original data file

A **map file** contains a data (vectors) classification. This arrangement is made of outstanding vectors, the code vectors. Each vector represents a classification class. In a map file, these code vectors are interrelated by a topology. There is also an additional information that associates the original data file with the classification. Each original vector might have been assigned to a code vector. To see that, for each code vector, there is a list of the indexes of the source data file original vectors . Since indexes are used, instead the vectors themselves, some operations over this file will be impossible without the original data file.

Fuzzy Codebook File. A codebook file contains a data (vectors) classification. This arrangement is made of outstanding vectors, the code vectors. Each vector represents a classification class. In a fuzzy codebook file, there is no relation between these code vectors. There is also an additional information, that associated the original data file with the classification. Each original vector might have been assigned to a code vector in a fuzzy mode. To show that, there is a membership matrix that includes the membership degree of each original data refer to each code vectors. Since there are references to the original data, some

operations over this file will be impossible without the original data file. To keep the compatibility with the standard codebook file, the list of indexes of original data is added, representing the maximum membership for each code vector.

Fuzzy Map File. A fuzzy map file contains a data (vectors) classification. This arrangement is made of outstanding vectors, the code vectors. Each vector represents a classification class. In a fuzzy map file, these code vectors are interrelated by a topology. There is also an additional information that associates the original data file with the classification. Each original vector might have been assigned to a code vector in a fuzzy mode. To show that, there is a membership matrix that includes the membership degree of each original data refer to each code vectors. Since there are references to the original data, some operations over this file will be impossible without the original data file. To keep the compatibility with the standard map file, the list of indexes of original data is added, representing the maximum membership for each code vector.

Distance Histogram File. The output of the Statistical Significance Procedure is an histogram with the data distance distribution. This file contains such an histogram.

Value Histogram File. The output of the Value Histogram Procedure is an histogram with the data distance distribution (real distances or randomise distances). This file contains such an histogram.

Hierarchical tree. A hierarchical tree file contains a data (vectors) classification in a hierarchical binary tree. It does not contain the original data, but their references. Many of the operations on hierarchical tree files, including visualization, will need the associated data set (file.dat)

Principal Components File (Main Features file). Principal components analysis is a quantitatively rigorous method for data reduction through the linear combination of dependent variables. All PCs are orthogonal to each other, so there is no redundant combination. This allows, for example, the projection of the original data set over a cartesian space. The Principal Components File contains the description of the PC factors.

Information file. These type of files contains information about the previous operations performed to obtain this file. This information includes in general, the process applied, its parameters, and so on.

Progress execution file. Progress files are temporary files; they store the current operation status. The progress is displayed by means of the current sub-operation name and a progress percentage. This percentage refers to the current sub-operation, not to the whole operation. The file name, without the .pro extension, will be the name of the operation outputs. A Progress file page is automatically refreshed.

Silhouette File. A silhouette file contains the silhouette value of each element. The silhouette value is a measure of the classification quality. These values lies between 1 and -1, where values near 1 represent a good classification; and values that fall under 0 are accepted as badly classified (in fact, this element is on average closer to members of some other cluster the one to which it is currently assigned. The silhouette values depend on how closed the elements of a cluster are between them and how far they are from the next closest cluster.

Sammon file. Sammon's mapping is an iterative method based on a gradient search (John W. Sammon, Jr. A nonlinear mapping for data structure analysis. IEEE Transactions on Computers, C-18(5):401-409, May 1969). The aim is to map points in n-dimensional space into a lower dimension (usually 2 dimensions). The basic idea is to arrange all the data points on a 2-dimensional plane in such a way, that the distances between the data points in this output plane

resemble the distances in vector space as defined by some metric as faithfully as possible and is thus useful for determining the shape of clusters and the relative distances between them.

Transactions File. This file contains a transactions set over which it is possible to run the "Association rule discovering" algorithm. It is a binary file with the following format :

<RowID, TransID, NumItems, List-Of-Items[Numitems] > *where* :
RowID: is the row identification
TransID: is a transaction identification
NumItems: is the number of elements in the transaction
List-Of-Items[NumItems] is the list of items
Each of these is a 4-byte integer.

Association Rules File. This file is generated upon a Transactions file, by means of the Association Rules Discovering procedure. It contains the rules that interrelate the different variables of a data file.

FIRST RULES

CONF	%SUPP	SUPP	IMPROV	ANTECEDENT	->	CONSEQUENT
100	0.227531	4	30.3103	[+]Col4,[+]Col10	->	[+]Col12
100	0.284414	5	12.7391	[+]Col9,[+]Col17,[+]Col20	->	[+]Col23
100	0.39818	7	12.7391	[+]Col17,[+]Col20,[+]Col22	->	[+]Col23
100	0.227531	4	15.4211	[+]Col10,[+]Col15,[+]Col16	->	[+]Col1
100	0.227531	4	22.8312	[+]Col1,[+]Col10,[+]Col16	->	[+]Col15
100	0.227531	4	10.1034	[+]Col15,[+]Col21,[+]Col22	->	[+]Col17
100	0.227531	4	15.4211	[+]Col13,[+]Col16,[+]Col22	->	[+]Col1
100	0.227531	4	15.4211	[+]Col13,[+]Col16,[+]Col23	->	[+]Col1
100	0.227531	4	14.4098	[+]Col1,[+]Col16,[+]Col23	->	[+]Col13

Bottom Page(s)

At the bottom the different operations that can be performed when a directory is selected, are displayed. These operations are :

Delete	**Delete Directory**. Only if it is not the home. Since the deleted directory is the current directory, after the operation is finished, the father directory is listed.
Refresh	**Refresh directory list**. Reloads the page, refreshing the progress values, the available free space, ...
Logout	**Logout**. Close the session and returns to the login Screen.
Rename	**Rename Directory.** The new name has to be specified first in the associated text field; then the Return key or the Rename button must be pushed. The result appears in a new page.
Create	**Create Directory.** The new name has to be specified first in the associated text field; then the Return key or the Create button must be pushed. The list is refreshed and the new directory appears.
Upload	**Upload a file.** Only data files can be loaded. To send a file to the server, the path file must be specified in the associated text field; the adjacent bottom can also be used. Then the **Upload** button must be pushed. This process may last several minutes. Whether the process has been successful, or not, the next page will be shown. Data files format is very specific, and it is explained in the data files page.

Data file

A data file contains a list of vectors (data), all of the same dimension (number of variables). Moreover, a file may contain some metadata, arranged in arrays labels, variables labels and global labels. A more detailed data file description is shown above at **Data File Format**.

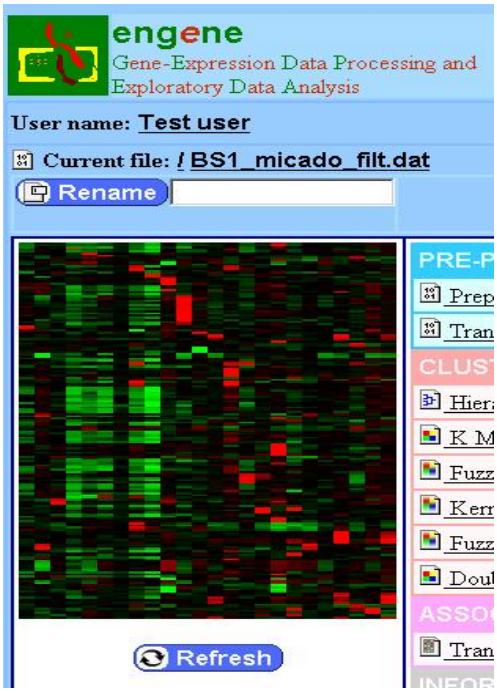
The **Data file** page shows the contents of the file. This file is owned by the user at **User name**. The **User name** links to the user home directory. The current directory path is displayed at **Current directory**. This path is organized into click-able subdirectories. On the right the file size is shown at **File size** and the file creation date at **File date**.

Viewer

On the left of the page there is an overview image with the data visual. This view is generated upon request. This means that it is formed the first time the data are selected. The view may take a few minutes to be created. Once the page has been completely refreshed, it will appears. To refresh the page you must press the refresh button (

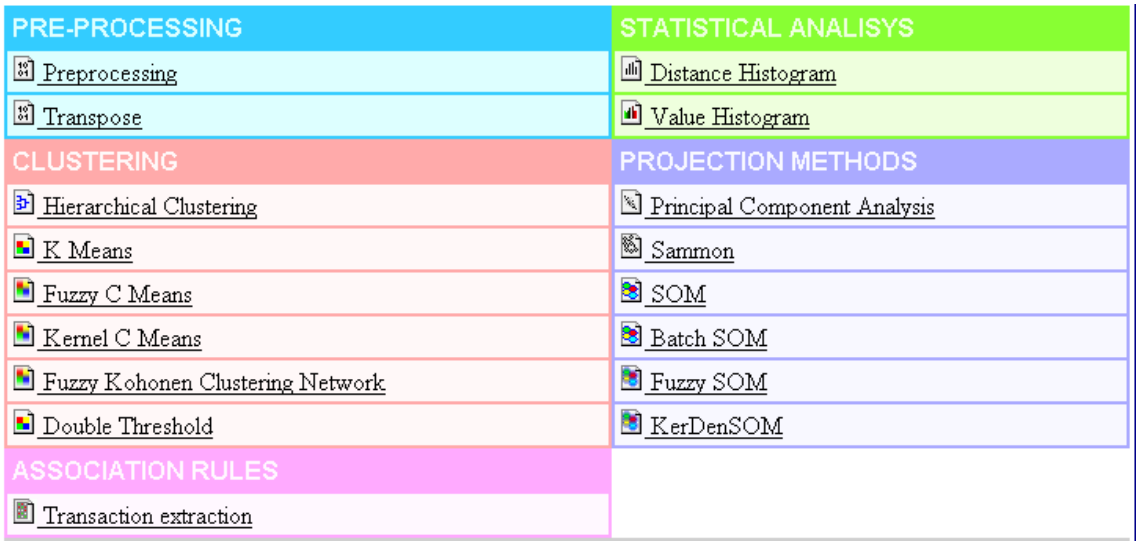
).

In the view the positive values are drawn in red, the negative values are drawn in green and the unknown values are drawn in grey. The view size is fixed, and if the amount of data is high, some of them may not be represented.



Operations

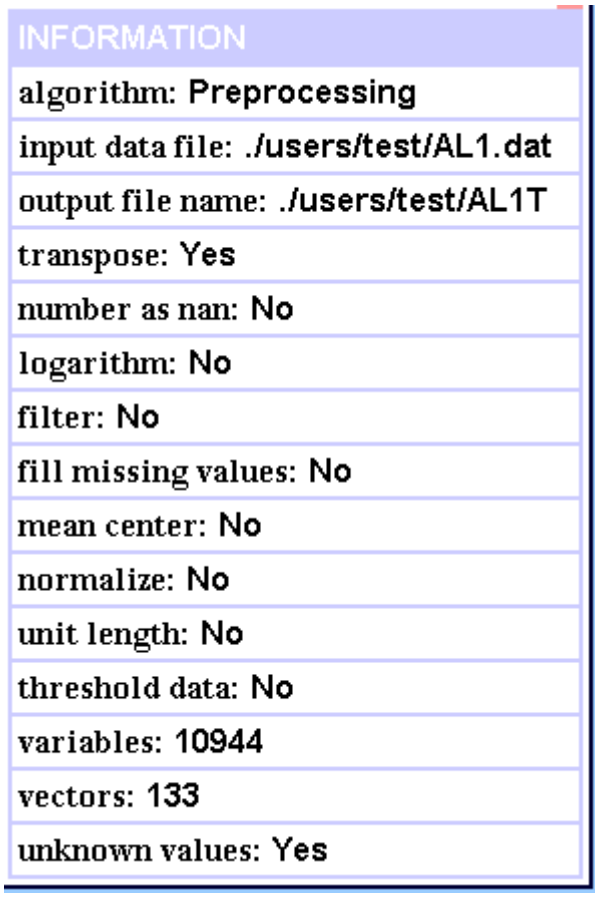
On the right of the image several operations are listed; all the different operations that a user can realize with the data. Any operation results into a file. The output files types are shown to the



left of the each operation name. Since there is a big assortment of operations, they are grouped according to what they do. First, there are the pre-processing operations (**Preprocessing**). The output of a these operations are modified data files. Then the analysis operations (**Analysis**) allows to generate statistical information or some other kind of information from the input data. The output will depend on the analysis type. Finally the clustering operations (**Clustering**) matches data, creating clusters according to specific criteria.

The available operations, their descriptions, and their links are listed below. A more detailed description about each option is available in the on-line help.

Output	Name	Short Description
	Preprocessing	Several pre-processing types frequently used, like filters, normalization, missing value filling, transformations
	Transpose	Interchanges columns and rows
	Hierarchical Clustering	Clusters data in pairs in a recursive form.
	K Means	Clusters data into K sets.
	Fuzzy K Means	Clusters data into K fuzzy sets.
	KCMeans	Kernel Density Estimator Clustering Algorithm.
	Fuzzy Kohonen Clustering	Fuzzy partition (clustering) using Fuzzy Kohonen Clustering Algorithm.
	Double Threshold	Clusters the nearest data for a given threshold and separates the farther data for another threshold.
	Transaction Extraction	Produces a set of transactions over which it is possible to apply association rules extraction procedure.
	Distance Histogram	Obtains the data distances distribution .
	Value histogram	Obtains the data values distribution.
	Principal component analysis	Searches for the data representation that most fits the data distribution.
	Sammon	Reduces the number of dimensions of data with no linear form.
	SOM	Clusters data by means of an auto-organized map.
	Batch SOM	Clusters data by means of an auto-organized map.
	Fuzzy SOM	Clusters data by means of a fuzzy auto-organized map.
	KerDenSOM	Kernel Probability Density Estimator Self-Organizing Map.



Information file

The data file information is shown under the operations. This type of files contains information about the previous operations performed to obtain the file related to it. This information includes in general, the process applied, its parameters, and so on. The Information file is also generated whenever an error occurs during the procedure execution. The output file supposed to be generated is not; in stead, there is an information file, with the same name as the output file should have, but with the extension .inf.

Some more information about the main options.

Preprocessing: Seldomly a data file is ready to be processed. Frequently, there are missing values (absent, unknown, ...), here called NaN, and also flat or low magnitude expression patterns can be found. Pre-processing tools supply a set of procedures to allow adjusting, filtering, filling, transposing and transforming original data sets, preparing them for a clustering procedures. Pre-processing procedures can combine in the same run several operations (filtering, Log-transforming, mean-centering, normalizing,...) which are executed in the order indicated by the parameters.

Transpose. Performs the traditional matrix transpose operation, that is to say, interchange rows and columns. This option has been include to allow large number of rows matrix (frequently used in the field) be transposed. The user should take note that in the following all the operations performed over these data must be properly interpreted.

Sammon. It is a non-linear mapping technique intended to map a set of high-dimensional input data into a lower dimensional space (usually 2) by trying to preserve the distances and local geometric relations of the original space.

Statistical Significance. Most of the time, without a knowledge of the input data, it is difficult to estimate correct values for the thresholds. When clusters generating, where distance thresholds are used, it is interesting to know the distribution of the distances between data. This is actually the purpose of the Statistical Significance.

Value Histogram. Most of the time, without a knowledge of the input data, it is difficult to estimate correct values for the thresholds. When using associative rules, where value threshold are used, it is interesting to know the distribution of the data values. This is actually the purpose of the Value Histogram

Principal Components (PC) are a linear combination of the original variables. All the PC are orthogonal to each other. The first PC is a single axis in space. When projecting data in that axis, the variance of these variables is the maximum among all the possible directions. In this way, it is easier to analyse data structure within a low number of dimension, generally the two dimensions of a screen or a sheet of paper.

K-Means It is one of the simplest clustering method. Some cluster centers are selected randomly, and then they are fine tuned in several iterations, using input data.

Double Threshold. This procedure puts together data whose distance is under a specific threshold, and separates them if the distance is above another specific threshold. It is a fast procedure, but the outputs may be poor. The two thresholds (upper and lower) are used in the following way: Data with distances under the *Lower threshold* belong to the same group and data with distance above the *Higher threshold* belong to different clusters. Data with distance between both threshold are compared with the current components of the group to take a decision.

Fuzzy K-Means. It is a standard clustering algorithm that Cluster data into K fuzzy sets.

Kernel C-means: Kernel Probability Density Estimating Clustering. It is a clustering algorithm based on kernel density estimator. For more information, please see the following reference: "A Novel Neural Network Technique for Analysis and Classification of EM Single-Particle Images" A. Pascual-Montano, L. E. Donate, M. Valle, M. Bárcena, R. D. Pascual-Marqui, J. M. Carazo, Journal of Structural Biology, Vol. 133, No. 2/3, Feb 2001, pp. 233-245

Hierarchical Clustering. This is an agglomerative hierarchical clustering method. These procedures select the two closest elements and group them to form a cluster, that in the following will be taken as an unique element. The procedure is repeated until all the elements are grouped into only one (the root) node.

Fuzzy Kohonen Clustering Network. It is a clustering algorithm that combine both, SOM and fuzzy methods producing very nice Self-Organizing properties.

Self Organizing Map. This procedure implements the well-known Kohonen Self-Organizing Map. It maps a set of high dimensional input vectors into a two-dimensional grid. For more theoretical information, please see the following reference: "Kohonen T. (1997) Self-Organizing maps, Second Edition, Springer-Verlag".

Batch SOM. This program implements the well-known Kohonen Self-Organizing Map using a training variant name "Batch training". It maps a set of high dimensional input vectors into a two-dimensional grid. For details see: "T. Kohonen, Self-Organizing Maps, Second Edition, Springer-Verlag (1997)". The BatchSOM algorithm uses several parameters which are described in the web-help page

Fuzzy Self Organizing Map. It maps a set of high dimensional input vectors into a two-dimensional grid using a fuzzy Self-Organizing Map. For more information, please see the following reference: "Smoothly Distributed Fuzzy *c*-Means: a New Self-Organizing Map.", Pascual-Marqui, R.D., Pascual-Montano, A., Kochi, K., Carazo, J.M., (2001). Pattern Recognition, 34, 2395-2402

Kernel Probability Density Estimator Self Organizing Map. It maps a set of high dimensional input vectors into a two-dimensional grid using a probabilistic neural network that select a set of code vectors that best resemble the probability density function of the original data. For more information, please see the following reference: “A Novel Neural Network Technique for Analysis and Classification of EM Single-Particle Images”, A. Pascual-Montano, L. E. Donate, M. Valle, M. Bárcena, R. D. Pascual-Marqui, J. M. Carazo, Journal of Structural Biology, Vol. 133, No. 2/3, Feb 2001, pp. 233-245

Association Rules

Note: this operations are in testing phase.

One of the most useful KDD (Knowledge Discovering and Data Mining) results (after Clustering) is in the form of association rules that make explicit the relationship between a set of antecedents and its associated consequents (i.e. the 89% of the customers that purchase bread and milk also purchase sugar). Additionally the significance of the rule can be assessed through its support (the percentage of transactions that contains the rule), the confidence (the percentage of transactions that containing the antecedents also contains the consequents) and the improvement (that indicates the enhancement of the rule's confidence compared to the statistical expectation).

A broad spectrum of algorithms for mining association rules has been developed from its introduction (Agrawal et al, 1993) with special attention to market basket data collections (Market Basket Analysis). We have developed a special algorithm "Transaction Driven Candidate Generation" to deal with data from the bioinformatic arena such as gene-expression data.

The association rule discovering algorithm works over a set of transactions. Thus the first step is to transform the gene-expression data (*.dat file type) into a transaction data file (*.tran file type). As result of this process a transaction file is obtained. Over this transaction file the "Association rule discovering" procedure can be applied

engene™, includes, at present, two operations to proceed in this field: production of the transactions set and, association rule discovering.

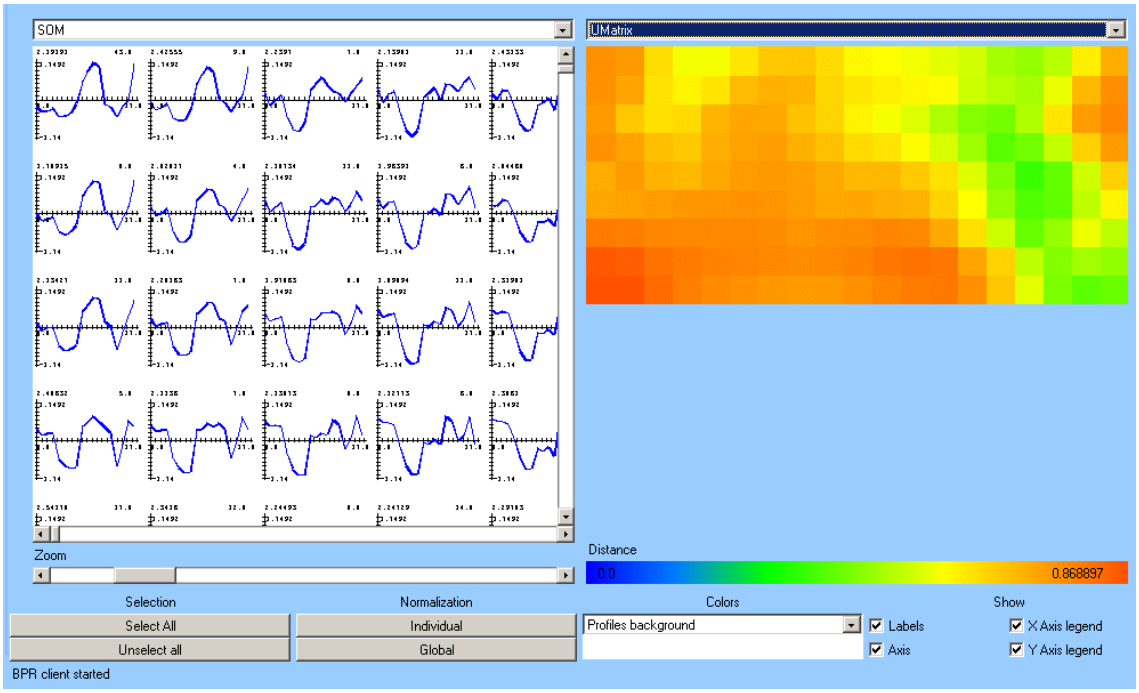
Transaction Extraction: Produces a set of transactions over which it is possible to apply association rules extraction procedure.

Association rule discovering procedure, which produce from the transaction set a collection of rule that correlate the expression/inhibition of specific genes with functional annotations corresponding to that genes

Java applet for visualizing Self-Organizing Maps

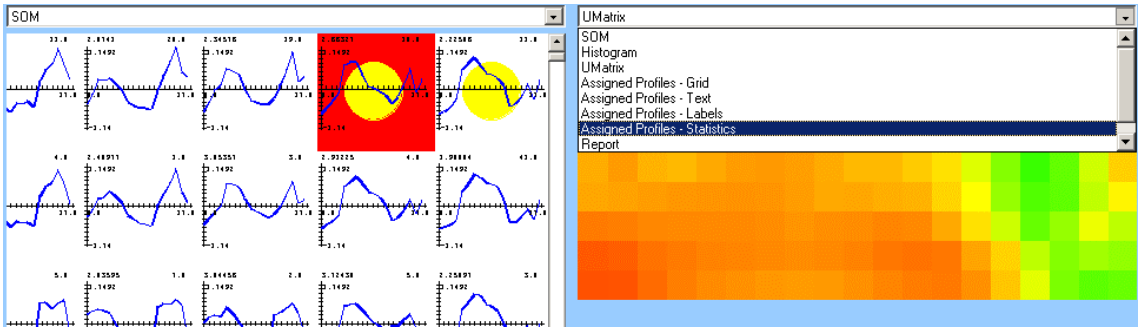
This java tool enables the interactive exploratory data analysis of self-organizing maps (SOMs). These mapping methods allow the projection of high-dimensional gene expression data into a lower dimensionality space in such a way that they can be efficiently explored and visualized to detect the clustering structure of the data set. With this applet, SOMs can be interactively explored, including a large set of options like histogram visualization, inter-neuron distance visualization (u-matrix), statistics of the clusters and others. In this way, the user can explore the data set using a reduced, but still informative set of representative units.

Once the applet is loaded with the SOM data, the following windows appears:



In the left pane, the self organizing units are displayed. They can be either zoomed in and zoomed out and completely browsed using the horizontal and vertical scroll bars. The profile information (colors, legends and labels) can also be customized using the options at the bottom of the page.

In addition, a large set of possibilities are available to extract information about the original expression profiles assigned to each code vector in the map:



The user can click on one or many code vectors in the map in order to select them and then go to the drop down menu at the right pane to select any of the following options:

Histogram*:* A color coded histogram is displayed, showing the number of original profiles assigned to each code vector.

UMatrix: Unified Distance Matrix. This option shows a colorful map that express the similarities among code vectors. Those homegenous areas represent similar zones or clusters in the map. It helps in identifying the clusters in the SOM.

Assigned profiles-Grid: When this option is selected, the original expression profiles assigned to the selected codevectors are shown.

Assigned profiles-Text: When this option is selected, the numerical expression values of the original expression profiles assigned to the selected codevectors are shown.

Assigned profiles-labels: When this option is selected, the meta data of the original expression profiles assigned to the selected codevectors is shown.

Assigned profiles-Statistics: When this option is selected, the mean and standard deviation of the original expression profiles assigned to the selected codevectors are shown.

Report: When this option is selected, a html report containing all the original expression profiles assigned to the selected codevectors is shown

Data File Format

A data file is a **table**. This table is stored in the file as a set of fields separated by **tab**, and along several lines. This text format may be worked out by Excel. So, an Excel table as follows will generate a file as shown below, when it is **saved as text**. this file is a data file in **engene**:

	A	B	C
1	1	16	72
2	123	15	1
3	151	15	23
4	32	516	53

```
1       16      72
123     15      1
151     15      23
32      516     53
```

Data are a collection of **vectors**, one vector a row. All vectors have the same number of **variables**, one variable a column. Some values may be unknown; in this case, the respective field may be a non numeric string o may be null. These values are called **NaN** (Not A Number). In next picture, these values are red marked.

	A	B	C
1	NAN	16	72
2	123	15	UNK
3	151		23
4	32	516	53

It is possible to append notes to data. This kind of information is called **metadata**. There are three types of metadata: **global labels**, **row labels** and **column labels**. All labels have two parts: the **label name** and the **label values**. For each global labels name there is only **one value**. Row labels have **one value for each data row**; and column labels have **one value for each data column**. Next picture shows how to put labels to data.

	A	B	C	D	E
1		CTag1Name	CTag1Val1	CTag1Val2	CTag1Val3
2		CTag2Name	CTag2Val1	CTag2Val2	CTag2Val3
3	RTag1Name				
4	RTag1Val1		1	16	72
5	RTag1Val2		123	15	1
6	RTag1Val3		151	15	23
7	RTag1Val4		32	516	53
8	GTag1Name	GTag1Val			
9	GTag2Name	GTag2Val			

Column labels names are red, and the values are yellow. Row labels names are green, and the values are blue. Global labels names are grey and the values are orange.

There must be a **space** between labels and data (yellow space in next figure). There must not have fields with value before the row and column labels names (in blue in next figure). And there must be nothing after the global labels (in green in next figure).

	A	B	C	D	E
1		CTag1Name	CTag1Val1	CTag1Val2	CTag1Val3
2		CTag2Name	CTag2Val1	CTag2Val2	CTag2Val3
3	RTag1Name				
4	RTag1Val1		1	16	72
5	RTag1Val2		123	15	1
6	RTag1Val3		151	15	23
7	RTag1Val4		32	516	53
8	GTag1Name	GTag1Val			
9	GTag2Name	GTag2Val			

Note : when working with Excel, you must mind the local configuration used to represent numbers; **engene** works with numbers with no thousands separation and uses a decimal point for decimal separator.

Other Data File Formats compatible with Engene:

Engene is also able to read and work with two other type of data files widely used in DNA Arrays analysis community:

- **Cluster software:**
  Cluster and TreeView are an integrated pair of programs for analyzing and visualizing the results of complex microarray experiments. Both written by Michael Eisen.(Eisen Lab: http://rana.lbl.gov/EisenSoftware.htm)
  This type of files need to have the **.clu** file extension in order to allow **engene** read it and convert it.
- **GeneCluster software:**
  GeneCluster was developed by Pablo Tamayo. It is a standalone Java application implementing the SOM algorithm.
  (http://www-genome.wi.mit.edu/cancer/software/genecluster2/gc2.html)
  This type of files need to have the **.res** file extension in order to allow **engene** read it and convert it.